CHAPITRE 1

ANALYSE D'ERREURS

1.1 REPRESENTATION DES NOMBRES DANS UN CALCULATEUR

Un calculateur ne peut fournir que des réponses approximatives.
Les approximations utilisées dépendent à la fois des contraintes physiques (espace mémoire ...) et du choix des méthodes retenues par le créateur du programme.
La première contrainte est que le système numérique d'un calculateur quelconque est discret. C'est-à-dire qu'il ne comporte qu'un nombre fini de nombres.
Il en découle que, sauf dans les cas les plus simples, tous les calculs seront entachés d'erreurs.

Exemple

Nous voulons calculer la valeur de $(11111111)^2$.
Sur une calculatrice, on trouve une valeur approchée comme $1.234567876543e + 14$
La valeur exacte est 123456787654321

1.1.1 Stockage des nombres

Les nombres sont stockés dans un ordinateur comme entiers ou réels.

– **Les nombres entiers**

Les nombres entiers sont ceux auquels nous sommes habitués sauf que le plus grand nombre représentable dépend des nombres d'octets utilisés.

- Avec deux octets, on peut représenter les entiers compris entre -32768 et 32767

- Avec quatre octets, on peut représenter les entiers compris entre -2147483648 et 2147483647

- **Les nombres réels**

Dans la mémoire d'un calculateur les nombres réels sont représentés en notation flottante.
En notation flottante, un nombre a la forme :

$$a = Y \times b^e$$

. b est **la base** du système numérique utilisé.
. Y est **la mantisse.**
. e est **l'exposant.**

- **La base**

La base est celle dans laquelle on écrit les nombres. (La plupart des calculatrices utilisent le système décimal c'est-à-dire $b = 10$. Les ordinateurs utilisent le système binaire c'est-à-dire $b = 2$.)

- **La mantisse**

La mantisse Y est un nombre de la forme :

$$Y = \pm 0.d_1 d_2 \cdots d_t$$

où $0 \leq d_i < b$ et t est fixé et fini avec $d_1 \neq 0$.
Les d_i sont les chiffres décimaux de Y (ou les digits).
Il faut, en général, un nombre infini de digits pour écrire exactement un nombre réel quelconque (on en a calculé des millions pour Π).
Dans une calculatrice, le nombre t est généralement voisin de 10.
Dans les grands ordinateurs, ce nombre peut prendre habituellement deux valeurs, la plus petite correspond à ce que l'on appelle la simple précision et la plus grande à la double précision.

- **L'exposant**

L'exposant e donne l'ordre de grandeur du nombre. On a :

$$m \leq e \leq M$$

où m et M sont des caractéristiques du calculateur.

Si $e > M$ on dit qu'il y a dépassement (ou surpassement) de capacité.

Si $e < m$ on dit qu'il y a sous passement de capacité.

Dans le cas d'un dépassement de capacité dans l'ordinateur il y a arrêt des calculs et l'impression d'un message d'erreur.

Dans le cas d'un sous passement de capacité certains ordinateurs s'arrêtent et imprime un message d'erreur tandis que d'autres remplacent le nombre en cause par zéro et continuent les calculs.

Si x est un nombre fourni à un ordinateur. On note $fl(x)$ sa représentation en virgule flottante.

<u>Troncature d'un nombre</u>

Considérons le nombre $x = \frac{1}{15} = 0.06666...$
Nous aurons si, $t = 5$

$$fl(x) = 0.66666 \times 10^{-1}$$

Nous avons tous simplement négligé les décimaux que nous ne pouvions stocker. On dit que l'on tronque et l'opération s'appelle **troncature.**

Par contre, dans les calculatrices, il y a la plupart du temps une opération **d'arrondissement.**

<u>Arrondissement d'un nombre</u>

Dans une représentation arrondie, lorsque la première décimale négligée est supérieure ou égale à 5, on ajoute 1 à la dernière décimale conservée.

Ainsi, pour $x = \frac{1}{15} = 0.06666...$ nous aurons, si $t = 5$

$$fl(x) = 0.66667 \times 10^{-1}$$

1.1.2 Erreur d'affectation

Théorème

Dans une arithmétique flottante à t digits on a :

$$| x - fl(x) | \leq 5 | x | p 10^{-t}$$

avec $p = 1$ dans le cas de l'arrondi et $p = 2$ dans le cas de la troncature.

1.2 OPERATIONS ARITHMETIQUES ELEMENTAIRES EN VIRGULE FLOTTANTE

L'addition et la soustraction flottante

- $\mathbf{x_1 \oplus x_2 = fl(fl(x_1) + fl(x_2))}$
- $\mathbf{x_1 \ominus x_2 = fl(fl(x_1) - fl(x_2))}$

Considérons x_1 et x_2 tels que : $| x_1 | \geq | x_2 |$
on a :
$fl(x_1) = 10^{e_1} Y_1$
$fl(x_2) = 10^{e_2} Y_2 = 10^{e_1} Y_2'$ avec $Y_2' = 10^{e_2 - e_1} Y_2$

Si les exposants ne sont pas les mêmes, on doit aligner, c'est-à-dire rendre le plus petit exposant égal au plus grand.

Exemples

On considère le cas $t = 4$

1. $fl(x_1) = 10^5 (0.4316)$ $\quad$ $fl(x_2) = 10^{-1} (0.1852)$
   On écrit : $fl(x_2) = 10^5 (0.0000001852)$
   $fl(x_1) + fl(x_2) = 10^5 (0.4316001852)$
   D'où : $x_1 \oplus x_2 = 10^5 (0.4316)$ en arrondissant ou en tronquant

2. $fl(x_1) = 10^5 (0.4316)$ $\quad$ $fl(x_2) = 10^2 (0.3422)$
   $fl(x_1) + fl(x_2) = 10^5 (0.4319422)$
   D'où : $x_1 \oplus x_2 = 10^5 (0.4319)$ en arrondissant ou en tronquant

3. $fl(x_1) = 10^5\,(0.4316) \qquad fl(x_2) = 10^5\,(0.7511)$
   $fl(x_1) + fl(x_2) = 10^5\,(1.1827) = 10^6\,(0.11827)$
   D'où : $x_1 \oplus x_2 = 10^6\,(0.1182)$ par troncature
   . $= 10^6\,(0.1183)$ par arrondi

4. $fl(x_1) = 10^5\,(0.4316) \qquad fl(x_2) = 10^5\,(0.4315)$
   $fl(x_1) - fl(x_2) = 10^5\,(0.0001)$
   D'où : $x_1 \ominus x_2 = 10^2\,(0.1000)$ en arrondissant ou en tronquant

La multiplication flottante

$$\mathbf{x_1 \otimes x_2 = fl(fl(x_1) \times fl(x_2))}$$

On a :

$$fl(x_1) \times fl(x_2) = 10^{e_1+e_2}\,Y_1\,Y_2$$

Remarquons que $0.1 \leq |\,Y_1\,|\,,|\,Y_2\,| < 1$ et que $0.01 \leq |\,Y_1\,Y_2\,| < 1$

Il sera par conséquent nécessaire, dans certains cas, de renormaliser la mantisse $Y_1\,Y_2$ afin que son premier digit soit non nul.

Exemples

1. $fl(x_1) = 10^2\,(0.2432) \qquad fl(x_2) = 10^3\,(0.2000)$
   $fl(x_1) \times fl(x_2) = 10^5\,(0.04864000)$
   D'où : $x_1 \otimes x_2 = 10^4\,(0.4864)$ par troncature ou arrondi

2. $fl(x_1) = 10^2\,(0.2432) \qquad fl(x_2) = 10^3\,(0.6808)$
   $fl(x_1) \times fl(x_2) = 10^5\,(0.16557056)$
   D'où : $x_1 \otimes x_2 = 10^5\,(0.1655)$ par troncature
   . $= 10^5\,(0.1656)$ par arrondi

La division flottante

$\mathbf{x_1 \oslash x_2 = fl(fl(x_1)/fl(x_2))}$

On a :
$\frac{fl(x_1)}{fl(x_2)} = 10^{e_1-e_2}\,\frac{Y_1}{Y_2}$

- si $\mid Y_1 \mid < \mid Y_2 \mid$ alors $0.1 \ < \mid \frac{Y_1}{Y_2} \mid < \ 1$. On calcule donc le quotient $\frac{Y_1}{Y_2}$ que l'on tronque ou l'on arrondit à t digits. L'exposant e est égal à $e_1 - e_2$.
- si $\mid Y_1 \mid \geq \mid Y_2 \mid$ alors $1 \ \leq \mid \frac{Y_1}{Y_2} \mid < \ 10$. On remplace alors Y_1 par $Y_1' = 10^{-1}\,Y_1$, on calcule $\frac{Y_1'}{Y_2}$ et l'on tronque ou l'on arrondi ensuite à t digits. L'exposant e est égal à $e_1 - e_2 + 1$.

Exemples

1. $fl(x_1) = 10^6\,(0.4323) \qquad fl(x_2) = 10^5\,(0.2000)$
   $\frac{fl(x_1)}{fl(x_2)} = 10^1\,(2.1615) = 10^2\,(0.21615)$
   D'où : $x_1 \oslash x_2 = 10^2\,(0.2161)$ par troncature
   . $= 10^2\,(0.2162)$ par arrondi
2. $fl(x_1) = 10^6\,(0.2539) \qquad fl(x_2) = 10^7\,(0.3000)$
   $\frac{fl(x_1)}{fl(x_2)} = 10^{-1}\,(0.8463333...)$
   D'où : $x_1 \oslash x_2 = 10^{-1}\,(0.8463)$ par troncature ou par arrondi

Associativité de l'addition

$x + (y + z)$ peut être différent de $(x + y) + z$ Soit, par exemple, à calculer la somme :

$$1 + 0.0004 + 0.0006 = 1.001$$

Avec un ordinateur pour lequel $t = 4$ en procédant par troncature. On a :

$1 \oplus 0.0004 = 1$
$(1 \oplus 0.0004) \oplus 0.0006 = 1$

$0.0004 \oplus 0.0006 = 0.001$
$1 \oplus (0.0004 \oplus 0.0006) = 1.001$

Cet exemple montre que l'addition flottante peut influencer le résultat de la sommation des séries à termes positifs.

Calcul de sommes à termes positifs, ordre de sommation

On veut calculer :

$$S = \sum_{i=1}^{n} a_i \qquad\qquad a_i > 0$$

En arithmétique finie, on calcule cette somme en formant la suite des sommes partielles :

$S_1 = fl(a_1)$
$S_2 = S_1 \oplus a_2$
$\vdots$
$S_k = S_{k-1} \oplus a_k \qquad k = 2, \cdots, n$

Le résultat est alors $S_n \simeq S$.

L'ordre dans lequel on somme les a_i peut changer la valeur de la somme S_n car l'arithmétique flottante n'est pas associative.

Exemple

Soit

$$S = 1 + \sum_{i=1}^{n} \frac{1}{i^2 + i} = 2 - \frac{1}{n+1}$$

Calculer cette somme, en arithmétique flottante, de deux façons différentes :

$$S_{n,1} = 1 + \frac{1}{2} + \cdots + \frac{1}{n^2 + n}$$
$$S_{n,2} = \frac{1}{n^2 + n} + \cdots + \frac{1}{2} + 1$$

avec $n = 9\,,\ 99\,,\ 999\,,\ 9999$

n	$S_{n,1}$	$S_{n,2}$	Valeur exacte S
9	1.9000000000	1.9000000000	1.9
99	1.9900000000	1.9900000000	1.99
999	1.9990000000	1.9990000000	1.999
9999	1.9998999999	1.9999000000	1.9999

On voit que l'on obtient des résultats différents selon que l'on somme de 1 à n ou de n à 1 ;les meilleurs résultats étant obtenus dans le second cas.

Perte de chiffres significatifs dans la soustraction

$\mathbf{x_1} \ominus \mathbf{x_2} = \mathbf{fl}(\mathbf{fl}(\mathbf{x_1}) - \mathbf{fl}(\mathbf{x_2}))$

Exemple

Considérons les nombres $\sqrt{7001}$ et $\sqrt{7000}$.
En arithmétique flottante à 8 chiffres, on a :

$$\sqrt{7001} \simeq 0.83671979 \times 10^2$$

$$\sqrt{7000} \simeq 0.83666003 \times 10^2$$

Donc

$\sqrt{7001} - \sqrt{7000} = fl((0.83671979 - 0.83666003) \times 10^2) = 0.59760000 \times 10^{-2}$

On peut obtenir un résultat plus précis en utilisant l'identité suivante :

$$\sqrt{x} - \sqrt{y} = (\sqrt{x} - \sqrt{y}) \times \frac{\sqrt{x} + \sqrt{y}}{\sqrt{x} + \sqrt{y}}$$

On obtient alors :

$1 \oslash (\sqrt{7001} \oplus \sqrt{7000}) = 1 \oslash (0.16733798 \times 10^3)$
$= 0.59759297 \times 10^{-2}$

La valeur exacte est $0.597592962824 \times 10^{-2}$

La soustraction est l'opération la plus dangereuse en calcul numérique. Elle peut amplifier l'erreur relative de façon catastrophique.

1.3 INSTABILITE NUMERIQUE

Définition

On dira que le calcul ou l'algorithme est **numériquement instable** si de petits changements dans les données entraînent de petits changements dans les résultats. Evidemment, dans le cas contraire on dira qu'il y a **instabilité numérique.**

Exemple d'instabilité numérique

On veut calculer

$$f(n) = \int_0^1 \frac{x^n}{a+x}\, dx \qquad\qquad a = cte > 1$$

Nous allons exprimer $f(n)$ récursivement :

$$\begin{aligned} f(n) &= \int_0^1 \frac{x^{n-1}\,(x+a-a)}{a+x}\, dx = \int_0^1 x^{n-1}\, dx - a \int_0^1 \frac{x^{n-1}}{a+x}\, dx \\ &= \frac{1}{n} - a\, f(n-1) \qquad\qquad n \geq 1 \end{aligned}$$

$f(0) = \mathrm{Ln}(\frac{1+a}{a})$

L'algorithme fourni par cette relation est numériquement instable.

Voici les résultats obtenus pour $a = 10$ et
$n = 0, 1, \cdots, 12$

n	$f(n)$ calculé	$f(n)$ exact
0	0.0953102	0.0953102
1	0.0468982	0.0468982
2	0.0310180	0.0310180
3	0.0231535	0.0231535
4	0.0184647	0.0184647
5	0.0153527	0.0153529
6	0.0131401	0.0131377
7	0.0114558	0.0114806
8	0.0104421	0.0101944
9	0.0066903	0.0091672
10	0.0330968	0.00832797
11	0.2400592	0.00762944
12	2.4839249	0.00703898

à partir de $n = 5$, les valeurs calculées sont de moins en moins précises à chaque itération ; pour $n \geq 10$ les résultats obtenus sont complètement erronés.

Cet algorithme est d'autant plus instable que a est plus grand que 1.
Pour ce faire supposons que l'erreur d'arrondi sur $f(0)$ est égale à ε_0 et qu'aucune erreur n'est introduite dans les calculs subséquats.

Notons $\widehat{f}(n)$ les valeurs calculées.

$\widehat{f}(0) = f(0) + \varepsilon_0$

$\widehat{f}(n) = \frac{1}{n} - a\,\widehat{f}(n-1) \qquad\qquad n = 1, 2, \cdots$

Par suite, si r_n désigne l'erreur sur $f(n)$

$$
\begin{aligned}
r_n = \widehat{f}(n) - f(n) &= -a\,\widehat{f}(n-1) + \frac{1}{n} - \frac{1}{n} + a\,f(n-1) \\
&= -a\,(\widehat{f}(n-1) - f(n-1)) \\
&= -a\,r_{n-1} \qquad\qquad n = 1, 2, \cdots
\end{aligned}
$$

et donc, puisque $r_0 = \varepsilon_0$, nous trouvons $r_n = (-a)^n\,\varepsilon_0$
$n = 1, 2, \cdots$

L'erreur initiale est multipliée par un facteur **a** à chaque itération.

CHAPITRE 2

INTERPOLATION POLYNOMIALE

La façon la plus simple d'approcher une fonction est l'utilisation d'un polynôme d'interpolation.
Le procédé est le suivant :

1. On choisit $n+1$ points distincts $x_0, x_1, \cdots, x_n$
2. On calcule $y_0 = f(x_0), y_1 = f(x_1), \cdots, y_n = f(x_n)$
3. On cherche un polynôme P_n de degré n tel que : $P_n(x_i) = y_i$, $i = 0, 1, \cdots, n$

Nous allons montrer l'existence et l'unicité d'un tel polynôme en le construisant effectivement.

2.1 Interpolation de Lagrange

Théorème et définition

Il existe un polynôme P_n de degré n et un seul, tel que :

$$P_n(x_i) = f(x_i) \quad \forall i = 0, 1, \cdots, n$$

Ce polynôme s'écrit : $P_n(x) = \displaystyle\sum_{i=0}^{n} f(x_i)\, L_i(x)$

avec $L_i(x) = \displaystyle\prod_{j=0, j\neq i}^{n} \frac{x - x_j}{x_i - x_j}$ (polynômes de Lagrange)

Le polynôme P_n s'appelle **le polynôme d'interpolation de Lagrange** de la fonction f relativement aux points $x_0, x_1, \cdots, x_n$.

Preuve.

- **unicité :**

  Supposons qu'il existe un polynôme P_n^* de degré n vérifiant : $P_n^*(x_i) = f(x_i) \quad \forall i = 0, 1, \cdots, n$

  Posons : $Q_n = P_n - P_n^*$.
  Q_n serait un polynôme de degré n vérifiant : $Q_n(x_i) = 0 \quad \forall i = 0, 1, \cdots, n$ de sorte que Q_n aurait au moins $(n+1)$ racines distinctes.
  Ceci n'est possible que si $Q \equiv 0$ (polynôme identiquement nul) ce qui prouve l'unicité de P_n.

- **existence :**

  Nous voyons immédiatement que : L_i est un polynôme de degré n et $L_i(x_j) = \delta_{i,j}$ (symbole de Kronecker).
  Il en résulte que le polynôme P_n est de degré n et vérifie : $P_n(x_i) = f(x_i) \quad \forall i = 0, 1, \cdots, n$

Remarque

Les $(n+1)$ polynômes de Lagrange sont linéairement indépendants et forment donc une base de l'espace vectoriel des polynômes de degré inférieur ou égal à n, appelée **base de Lagrange**.

Calcul de $P_n(\alpha)$

Le calcul de $P_n(\alpha)$ nécessite celui des $(n+1)$ quantités $L_i(\alpha)$, ce qui est coûteux.

La méthode de Lagrange est d'un intérêt plus théorique que pratique car :

- coûteux en nombres d'opérations
- formulation peu aisée : si l'on ajoute un point x_{n+1} les L_i doivent entièrement être recalculés.

2.2 Erreur d'interpolation

Etude de l'erreur $\mathbf{e_n(x) = f(x) - P_n(x)}$ (appelée **erreur d'interpolation**) pour tout $x \in [a, b]$

Théorème

Si $f \in \mathcal{C}^{(n+1)}([a,b])$ alors pour tout $x \in [a,b]$, il existe ξ_x appartenant au plus petit intervalle fermé I contenant $x, x_0, x_1, \cdots, x_n$ tel que :

$e_n(x) = f(x) - P_n(x) = F(x)\,\frac{f^{(n+1)}(\xi_x)}{(n+1)!} \qquad (*)$

où $F(x) = (x - x_0)\,(x - x_1)\,\cdots(x - x_n)$

Preuve.

Si $x = x_i$ alors $e_n(x) = 0$ et l'égalité (*) est vérifiée trivialement ;
Supposons que $x \neq x_i \quad \forall i = 0, 1, \cdots, n$ et considérons pour x fixé la fonction g définie par :

$g(t) = e_n(t) - \frac{F(t)}{F(x)}\,e_n(x)$

La fonction $g \in \mathcal{C}^{(n+1)}([a,b])$ et s'annule en $(n+2)$ points distincts $x, x_0, x_1, \cdots, x_n$.
Le théorème de Rolle montre que g' admet au moins $(n+1)$ racines dans I.
D'où, en procédant par récurrence sur l'ordre de dérivation de g, la fonction $g^{(n+1)}$ admet au moins une racine dans I. Soit ξ_x cette racine. On a :

$0 = g^{(n+1)}(\xi_x) = f^{(n+1)}(\xi_x) - \frac{(n+1)!}{F(x)}\,e_n(x)$

D'où : $e_n(x) = F(x)\,\frac{f^{(n+1)}(\xi_x)}{(n+1)!}$

2.3 Interpolation d'Hermite

Problème : Déterminer un polynôme qui coïncide avec f, ainsi que sa dérivée avec f', aux points $x_0, x_1, \cdots, x_n$.

Théorème et définition

Etant donnée une fonction f définie sur [a,b] et admettant des dérivées aux points x_i, il existe un polynôme $P_{2\,n+1}$ de degré $2\,n+1$ et un seul tel que

$$P_{2\,n+1}(x_i) = f(x_i) \;\; et \;\; P'_{2\,n+1}(x_i) = f'(x_i)$$

$$\forall i = 0, 1, \cdots, n$$

Le polynôme P_{2n+1} ainsi défini est appelé **polynôme d'interpolation d'Hermite** de la fonction f relativement aux points $x_0, x_1, \cdots, x_n$.

2.4 Erreur d'interpolation

Etude de l'erreur $\mathbf{e_n}(\mathbf{x}) = \mathbf{f}(\mathbf{x}) - \mathbf{P_n}(\mathbf{x})$ pour tout $x \in [a, b]$

Théorème

On considère le polynôme d'interpolation d'Hermite P_{2n+1}.
On suppose que $f \in \mathcal{C}^{(2n+2)}([a, b])$ alors pour tout $x \in [a, b]$, il existe ξ_x appartenant au plus petit intervalle fermé I contenant $x, x_0, x_1, \cdots, x_n$ tel que

$$e_n(x) = f(x) - P_{2n+1}(x) = F^2(x) \frac{f^{(2n+2)}(\xi_x)}{(2n+2)!}$$

Preuve.

On considère pour tout x fixé distinct des x_i la fonction g de la variable t défini par

$$g(t) = e_n(t) - \frac{F^2(t)}{F^2(x)} e_n(x)$$

La fonction $g' \in \mathcal{C}^{(2n+1)}([a, b])$ et admet au moins $(2n + 2)$ zéros distincts.
Le théorème de Rolle montre que g'' admet au moins $(2n + 1)$ racines dans I.
D'où, en procédant par récurrence sur l'ordre de dérivation de g, la fonction $g^{(2n+2)}$ admet au moins une racine dans I. Soit ξ_x cette racine. On a :

$$0 = g^{(2n+2)}(\xi_x) = f^{(2n+2)}(\xi_x) - \frac{(2n+2)!}{F^2(x)} e_n(x)$$

D'où : $e_n(x) = F^2(x) \frac{f^{(2n+2)}(\xi_x)}{(2n+2)!}$

2.5 Interpolation itérée

La détermination et l'évaluation du polynôme de Lagrange sont assez coûteuses lorsque le nombre de noeuds s'accroît.
Nous allons développer un procédé itératif permettant de calculer le polynôme d'interpolation $P_n(x)$ basé sur $n + 1$ noeuds $x_0, x_1, \cdots, x_n$ à partir du polynôme $P_{n-1}(x)$ basé sur n noeuds $x_0, x_1, \cdots, x_{n-1}$.

Pour $n \geq 1$, le polynôme $P_n(x) - P_{n-1}(x)$ est de degré n et s'annule aux points $x_0, x_1, \cdots, x_{n-1}$, il est donc de la forme :

$P_n(x) - P_{n-1}(x) = a_n\,(x - x_0)\,(x - x_1)\,\cdots(x - x_{n-1})$

où a_n est le coefficient dominant de $P_n(x)$

D'après la formule de Lagrange on a :

$$P_n(x) = \sum_{k=0}^{n} f(x_k)\,L_k(x)$$

$L_k(x)$ est un polynôme de degré n dont le coefficient dominant est :

$\frac{1}{(x_k - x_0)\cdots(x_k - x_{k-1})\,(x_k - x_{k+1})\cdots(x_k - x_n)}$

donc

$$a_n = \sum_{k=0}^{n} \frac{f(x_k)}{(x_k - x_0)\,\cdots(x_k - x_{k-1})\,(x_k - x_{k+1})\cdots(x_k - x_n)}$$

Le nombre a_n est appelé **différence divisée** d'ordre n de la fonction $f(x)$ et est noté : $\mathbf{a_n = f[x_0, x_1, \cdots, x_n]}$

Définissant $f(x_0) = f[x_0]$, nous avons :

$P_n(x) = P_{n-1}(x) + (x - x_0)\,\cdots(x - x_{n-1})\,f[x_0, x_1, \cdots, x_n]$

En utilisant ces relations pour $n = 1, 2, \cdots$, nous pouvons donc :

1. Obtenir une représentation explicite du polynôme d'interpolation :

   $$\mathbf{P_n(x) = f[x_0] + \sum_{k=1}^{n} f[x_0, x_1, \cdots, x_k]\,(x - x_0)\,\cdots(x - x_{k-1})}$$

   (Cette représentation est appelée **représentation de Newton du polynôme**)

2. Calculer itérativement les valeurs $P_0(x), P_1(x), \cdots, P_n(x)$ en un point x donné

Si nous définissons :

$$f[x_i, x_{i+1}, \cdots, x_j] = \sum_{k=i}^{j} \frac{f(x_k)}{(x_k - x_i)\,\cdots(x_k - x_{k-1})\,(x_k - x_{k+1})\cdots(x_k - x_j)}$$

nous pouvons montrer que :

$f[x_i, x_{i+1}, \cdots, x_j] = \frac{f[x_{i+1},x_{i+2},\cdots,x_j]-f[x_i,x_{i+1},\cdots,x_{j-1}]}{x_j-x_i}$

Cette relation suggère de calculer les différences divisées, à l'aide d'une table triangulaire, de la façon suivante :

x_i	$f(x_i) = f[x_i]$	$f[x_i, x_{i+1}]$	$f[x_i, x_{i+1}, x_{i+2}] \cdots$
x_0	$f[x_0]$		
x_1	$f[x_1]$	$f[x_0, x_1]$	
x_2	$f[x_2]$	$f[x_1, x_2]$	$f[x_0, x_1, x_2]$
$\vdots$			

Exemple

Déterminer le polynôme P_3 d'interpolation de Lagrange de f aux points $(x_i, f(x_i))$ suivants : (0,1),(1,3),(3,2),(4,5)

x_i	$f[x_i]$			
0	1			
1	3	2		
3	2	$-\frac{1}{2}$	$-\frac{5}{6}$	
4	5	3	$\frac{7}{6}$	$\frac{1}{2}$

Le polynôme P_3 est donc donné par :

$P_3(x) = 1 + 2\,x - \frac{5}{6}\,x\,(x-1) + \frac{1}{2}\,x\,(x-1)\,(x-3)$

Soit en développant l'expression de P_3 dans la base canonique :

$P_3(x) = \frac{1}{2}\,x^3 - \frac{17}{6}\,x^2 + \frac{13}{3}\,x + 1$

Si on ajoute un point d'interpolation, il suffit de compléter le tableau des différences divisées

x_i	$f[x_i]$				
0	1				
1	3	2			
3	2	$-\frac{1}{2}$	$-\frac{5}{6}$		
4	5	3	$\frac{7}{6}$	$\frac{1}{2}$	
2	-1	3	0	$-\frac{7}{6}$	$-\frac{5}{6}$

$P_4(x) = P_3(x) - \frac{5}{6}\,x\,(x-1)\,(x-3)\,(x-4)$

$\quad = -\frac{5}{6}\,x^4 + \frac{43}{6}\,x^3 - \frac{56}{3}\,x^2 + \frac{43}{3}\,x + 1$

Corollaire

– L'erreur d'interpolation de Lagrange est donnée par

$e_n(x) = f(x) - P_n(x) = F(x)\, f[x_0, \cdots, x_n, x]$

– Si $f \in \mathcal{C}^{(n+1)}[a, b]$ alors il existe $\xi_x \in [a, b]$ tel que :

$f[x_0, \cdots, x_n, x] = \frac{f^{(n+1)}(\xi_x)}{(n+1)!}$

Preuve

Pour x fixé, le polynôme $Q(t)$ d'interpolation de Lagrange de f aux points $x_0, \cdots, x_n, x$ s'écrit :

$$Q(t) = P_n(t) + f[x_0, \cdots, x_n, x]\,(t - x_0) \cdots (t - x_n)$$

– Pour $t = x$, il vient

$f(x) = Q(x) = P_n(x) + F(x)\, f[x_0, \cdots, x_n, x]$

D'où la première relation

– la deuxième relation du corollaire découle immédiatement de la première.

Calcul de $\mathbf{P_n}(\alpha)$, $\alpha \in [\mathbf{a}, \mathbf{b}]$

$$P_n(\alpha) = f[x_0] + f[x_0, x_1]\,(\alpha - x_0) + f[x_0, x_1, x_2]\,(\alpha - x_0)\,(\alpha - x_1) \cdots + f[x_0, x_1, \cdots, x_n]\,(\alpha - x_0)\,(\alpha - x_1)\,(\alpha - x_{n-1})$$

Cette relation se réecrit de la manière suivante :

$$P_n(\alpha) = f[x_0] + (\alpha - x_0)\,(f[x_0, x_1] + (\alpha - x_1)\,(f[x_0, x_1, x_2] + \cdots (\alpha - x_{n-2})\,(f[x_0, x_1, \cdots x_{n-1}] + (\alpha - x_{n-1})\,(f[x_0, x_1, \cdots x_n]) \cdots)).$$

L'évaluation se fait de droite à gauche de la manière suivante :

ALGORITHME DE HÖRNER

$b_0 = f[x_0, x_1, \cdots x_n]$

$b_1 = f[x_0, x_1, \cdots x_{n-1}] + (\alpha - x_{n-1})\, b_0$

$\vdots$

$b_i = f[x_0, x_1, \cdots x_{n-i}] + (\alpha - x_{n-i})\, b_{i-1}$

$$i = 1, \cdots, n-1$$

$\vdots$

$b_n = f[x_0] + (\alpha - x_0)\, b_{n-1} = P_n(\alpha)$

Il est facile de vérifier que le coût opératoire pour calculer $P_n(\alpha)$ est de :

1. Algorithme de Newton : $n\,(n+1)$ soustractions

   $\frac{n\,(n+1)}{2}$ divisions

   $\frac{n\,(n+1)}{2}$ divisions

2. Algorithme de Hörner : n additions

   n soustractions

   n multiplications

Soit au total : $n^2 + 3\,n$ add/sous, n mult, $\frac{n\,(n+1)}{2}$ div

CHAPITRE 3

INTEGRATION NUMERIQUE

Notations

$\mathbf{P}_n$: ensemble des polynômes à une variable, à coefficients réels, de degré inférieur ou égal à n.

$[a, b]$: intervalle fini.

f : une fonction numérique définie et continue sur [a,b].

$\{x_0, x_1, \cdots, x_n\}$: $n+1$ points distincts de [a,b].

Ω : une classe de fonctions définies sur [a,b].

Problème : Déterminer des **formules de quadrature** de la forme :

$$I = \int_a^b f(x)\,dx = \sum_{i=0}^{n} A_i\, f(x_i) + E_n(f) \qquad (\star)$$

où $\sum_{i=0}^{n} A_i\, f(x_i)$ est une **valeur approchée** de l'intégrale I.

$E_n(f)$ **l'erreur de quadrature** associée

avec les paramètres $(A_i, x_i)_{0\leq i\leq n}$ calculés de manière que la formule $(\star)$ soit exacte sur Ω, c'est-à-dire :

$\forall f \in \Omega\ ,\ E_n(f) = 0$

3.1 Méthodes de Newton-Cotes

1. **Construction**

Les noeuds $\{x_i\}_{0\leq i\leq n}$ étant choisis équidistants ($h = \frac{b-a}{n}$, $x_i = a + i\,h$, $i = 0,\cdots,n$).

Remplaçons f par son polynôme d'interpolation de Lagrange $p_n(x)$ relativement aux points $x_0, x_1, \cdots, x_n$.

De la relation :

$\forall x \in [a,b]$, $f(x) = p_n(x) + e_n(x)$

où $e_n(x)$ représente l'erreur d'interpolation, il vient

$$I = \int_a^b f(x)\,dx = \int_a^b p_n(x)\,dx + \int_a^b e_n(x)\,dx \qquad (\star\star)$$

On a : $p_n(x) = \sum_{i=0}^{n} f(x_i)\,L_i(x)$ où $\{L_i\}_{0\leq i\leq n}$ désigne la base des polynômes de Lagrange de $\mathbf{P}_n$. La relation se réecrit :

$$I = \int_a^b f(x)\,dx = \sum_{i=0}^{n} A_i^{(n)}\,f(x_i) + E_n(f)$$

avec

$$A_i^{(n)} = \int_a^b L_i(x)\,dx\ ,\ i = 0,1,\cdots,n \text{ et}$$

$$E_n(f) = \int_a^b e_n(x)\,dx$$

La formule $(\star\star)$ de quadrature est exacte sur $\mathbf{P}_n$ car si $f \in \mathbf{P}_n$, alors $f = p_n$ en conséquence $E_n(f) = 0$.

2. **Calcul des poids**

Les coefficients ou poids $A_i^{(n)}$ de la formule de quadrature $(\star\star)$ sont données par le

Théorème

On prend : $x_i = a + i\,h$, $i = 0,1,\cdots,n$ avec $h = \frac{b-a}{n}$ une subdivision de l'intervalle [a,b] en n sous-intervalles égaux.

$\forall i = 0,1,\cdots,n$ on a :

$$\mathbf{A_i^{(n)} = A_{n-i}^{(n)} = \frac{(-1)^{n-i}\,h}{i!\,(n-i)!}\int_0^n \prod_{j=0_{j\neq i}}^{n} (u-j)\,du}$$

Preuve.

On a : $A_i^{(n)} = \int_a^b L_i(x)\,dx$ avec $L_i(x) = \prod_{j=0_{j\neq i}}^{n} \frac{x-x_j}{x_i-x_j}$

On en déduit que :

$$A_i^{(n)} = \frac{1}{\prod_{j=0_{j\neq i}}^{n} (x_i - x_j)} \int_a^b \prod_{j=0_{j\neq i}}^{n} (x-x_j)\,dx$$

On a :

$$\begin{aligned}\prod_{j=0_{j\neq i}}^{n} (x_i - x_j) = \prod_{j=0}^{i-1}(x_i-x_j) \prod_{j=i+1}^{n} (x_i-x_j) \\ &= (i!\,h^i)\,((-1)^{n-i}\,(n-i)!\,h^{n-i}) \\ &= (-1)^{n-i}\,i!\,(n-i)!\,h^n\end{aligned}$$

En faisant le changement de variable $u = \frac{x-a}{h}$ dans l'intégrale, on trouve :

$$\begin{aligned}\int_a^b \prod_{j=0_{j\neq i}}^{n} (x-x_j)\,dx \\ &= \int_0^n \prod_{j=0_{j\neq i}}^{n} (a+u\,h-a-j\,h)\,h\,du \\ &= h^{n+1} \int_0^n \prod_{j=0_{j\neq i}}^{n} (u-j)\,du\end{aligned}$$

Pour établir la symétrie des coefficients, on fait le changement de variable : $v = n-u$ dans $A_{n-i}^{(n)} = \frac{(-1)^i\,h}{(n-i)\,!i!} \int_0^n \prod_{j=0_{j\neq i}}^{n} (u-j)\,du$

il vient

$$\begin{aligned}\int_0^n \prod_{j=0_{j\neq i}}^{n} (u-j)\,du = -\int_n^0 \prod_{j=0_{j\neq i}}^{n} (n-j-v)\,dv \\ &= (-1)^n \int_0^n \prod_{j=0_{j\neq i}}^{n} (v-j)\,dv\end{aligned}$$

d'où : $A_{n-i}^{(n)} = A_i^{(n)}$ puisque $(-1)^{n+i} = (-1)^{n-i}$

Exemple 1 : $n = 1$ **Formule des Trapèzes**

$$A_0^{(1)} = A_1^{(1)} = h \int_0^1 u\, du = \frac{h}{2}$$

donc

$$\sum_{i=0}^{1} A_i^{(1)} f(x_i) = \frac{b-a}{2} [f(a) + f(b)]$$

Exemple 2 : $n = 2$ **Formule de Simpson**

$$A_0^{(2)} = A_2^{(2)} = \tfrac{h}{2} \int_0^2 (u-1)(u-2)\, du = \frac{h}{3}$$

$$A_1^{(2)} = -h \int_0^2 u(u-2)\, du = \frac{4h}{3}$$

donc

$$\sum_{i=0}^{2} A_i^{(2)} f(x_i) = \frac{b-a}{6} [f(a) + 4f(\frac{a+b}{2}) + f(b)]$$

3. **Erreur d'intégration**

– **Pour la formule des Trapèzes**

Pour évaluer le terme d'erreur pour la formule des Trapèzes, nous commençons par le lemme suivant

lemme

Soit $f \in \mathcal{C}^{(2)}([a,b])$. $\forall x \in [a,b]$, $\exists \xi_x \in [a,b]$ tel que :

$$f(x) = f(a) + (x-a)\,\tfrac{f(b)-f(a)}{b-a}$$
$$-\tfrac{1}{2}(x-a)(b-x)\, f"(\xi_x)$$

De plus l'application $x \to f"(\xi_x)$ est continue sur [a,b].

Preuve.

On pose : $\Phi(x) = f(x) - f(a) - (x-a)\,\tfrac{f(b)-f(a)}{b-a}$

$$-c(x-a)(b-x)$$

Soit $x_0 \in]a, b[$.

On choisit c tel que : $\Phi(x_0) = 0$

La fonction $\Phi \in \mathcal{C}^{(2)}([a, b])$ et s'annule en a, b, x_0.

Le théorème de Rolle montre que Φ' admet au moins 2 racines dans [a,b].
D'où la fonction $\Phi"$ admet au moins une racine dans [a,b]. Soit ξ_{x_0} cette racine.
On a :

$0 = \Phi"(\xi_{x_0}) = f"(\xi_{x_0}) + 2\,c$

d'où $c = -\frac{1}{2}\, f"(\xi_{x_0})$.

Il est immédiat de voir que : $x \to f"(\xi_x)$ est continue pour tout $x \neq a\ et\ x \neq b$.

– Pour $x \to a^+$

$$\lim_{x \to a^+} -\frac{1}{2}\, f"(\xi_x) = \frac{f'(a) - \frac{f(b)-f(a)}{b-a}}{b-a} \quad finie$$

– Pour $x \to b^-$

$$\lim_{x \to b^-} -\frac{1}{2}\, f"(\xi_x) = \frac{-f'(b) + \frac{f(b)-f(a)}{b-a}}{b-a} \quad finie$$

D'après la règle de l'Hospital, on peut donc prolonger $x \to f"(\xi_x)$ par continuité en a et en b.

Formule des Trapèzes :

si $\mathbf{f} \in \mathcal{C}^{(\mathbf{2})}([\mathbf{a}, \mathbf{b}])$ **avec** $\mathbf{h} = \mathbf{b} - \mathbf{a}$

$$\int_{\mathbf{a}}^{\mathbf{b}} \mathbf{f}(\mathbf{x})\, \mathbf{dx} = \frac{\mathbf{h}}{\mathbf{2}}\, [\mathbf{f}(\mathbf{a}) + \mathbf{f}(\mathbf{b})] - \frac{\mathbf{h}^3}{\mathbf{12}}\, \mathbf{f}^{(\mathbf{2})}(\xi), \qquad \xi \in [\mathbf{a}, \mathbf{b}]$$

Preuve.

$$\begin{aligned} E_1(f) &= \int_a^b f(x)\, dx - \frac{b-a}{2}\, (f(a) + f(b)) \\ &= \int_a^b (f(x) - f(a) - (x-a)\, \frac{f(b)-f(a)}{b-a})\, dx \\ &= -\tfrac{1}{2} \int_a^b f^{(2)}(\xi_x)\, (x-a)\, (b-x)\, dx \end{aligned}$$

La fonction $x \to (x-a)\,(b-x)$ a un signe constant sur [a,b] et $x \to f^{(2)}(\xi_x)$ est

continue. En appliquant donc la formule de la moyenne sous forme intégrale, on trouve :

$$E_1(f) = -\tfrac{1}{2}\, f^{(2)}(\xi) \int_a^b (x-a)\,(b-x)\,dx$$
$$= -\tfrac{1}{12}\, f^{(2)}(\xi)\,(b-a)^3$$

– **Pour la formule de Simpson**

Un calcul plus compliqué que pour la formule des Trapèzes donne l'évaluation de l'erreur $E_2(f)$

$$E_2(f) = -\tfrac{1}{90}\, f^{(4)}(\xi)\,(b-a)^5$$

Formule de Simpson :

si $\mathbf{f} \in \mathcal{C}^{(\mathbf{4})}([\mathbf{a},\mathbf{b}])$ **avec** $\mathbf{h} = \frac{\mathbf{b-a}}{\mathbf{2}}$

$$\int_{\mathbf{a}}^{\mathbf{b}} \mathbf{f(x)dx} = \frac{\mathbf{h}}{\mathbf{3}}[\mathbf{f(a)} + \mathbf{4\,f}(\frac{\mathbf{a+b}}{\mathbf{2}}) + \mathbf{f(b)}] - \frac{\mathbf{h^5}}{\mathbf{90}}\mathbf{f^{(4)}}(\xi),$$
$$\xi \in [\mathbf{a},\mathbf{b}]$$

Preuve. (admise)

Exemple : $I = \displaystyle\int_0^{\frac{\pi}{2}} sin(x)\,dx = 1$

Trapèzes :

$$I = \tfrac{\pi}{4}\,(sin(0) + sin(\tfrac{\pi}{2})) + \tfrac{(\frac{\pi}{2})^3}{12}\, sin(\xi)$$

$$I = \tfrac{\pi}{4} + \tfrac{\pi^3}{96}\, sin(\xi) = 0.78539 + 0.32298\, sin(\xi)$$

$$\xi \in [\mathbf{0}, \tfrac{\pi}{\mathbf{2}}]$$

Simpson :

$$I = \tfrac{\pi}{12}\,(sin(0) + 4\,sin(\tfrac{\pi}{4}) + sin(\tfrac{\pi}{2})) - \tfrac{(\frac{\pi}{4})^5}{90}\, sin(\xi)$$

$I = \frac{\pi}{12}(1+2\sqrt{2}) - \frac{\pi^5}{92160}\, sin(\xi)$

$= 1.00227 - 3.32052 \times 10^{-3}\, sin(\xi)$

$\xi \in [\mathbf{0}, \frac{\pi}{\mathbf{2}}]$

3.2 Méthodes de Newton-Cotes composites

Les formules de Newton-Cotes sont peu adaptées au calcul d'intégrales sur un grand intervalle d'intégration

Exemple :

$\int_0^4 e^x dx = e^4 - e^0 = 53.598..$

La valeur approchée obtenue par Simpson est :

$\frac{2}{3}\,(e^0 + 4\,e^2 + e^4) = 56.769$ ce qui est loin de la valeur exacte.

On peut penser à augmenter n de telle façon à diminuer le pas d'intégration $h = \frac{b-a}{n}$ mais cette solution s'avère numériquement mauvaise.

En effet, pour les formules de Newton-Cotes de degré n :

- Le calcul des $A_i^{(n)}$ devient difficile quand n augmente.
- L'erreur de quadrature $E_n(f)$ ne tend pas nécessairement vers 0, quand n tend vers l'infini pour une fonction donnée $f \in \mathcal{C}([a,b])$.

Il est préférable de subdiviser l'intervalle d'intégration en un grand nombre de petits sous-intervalles, et d'appliquer sur chacun d'eux la méthode des Trapèzes ou de Simpson.

On considère une subdivision de l'intervalle [a,b] en N intervalles égaux :

$t_i = a + i\,h$, $i = 0, 1, \cdots, N$ et $h = \frac{b-a}{N}$

On décompose :

$I = \int_a^b f(x)dx = \sum_{i=1}^{N} \int_{t_{i-1}}^{t_i} f(x)dx$

Chaque intégrale $\int_{t_{i-1}}^{t_i} f(x)dx$ étant calculée par une formule de Newton-Cotes de bas

degré.

- **Formule des Trapèzes composites :**

si $\mathbf{f} \in \mathcal{C}^{(2)}([\mathbf{a}, \mathbf{b}])$ **avec** $\mathbf{h} = \frac{\mathbf{b}-\mathbf{a}}{\mathbf{N}}$

$$\int_{\mathbf{a}}^{\mathbf{b}} \mathbf{f}(\mathbf{x})\, \mathbf{dx} = \frac{\mathbf{h}}{\mathbf{2}} \left[\mathbf{f}(\mathbf{a}) + \mathbf{2} \sum_{\mathbf{i}=\mathbf{1}}^{\mathbf{N}-\mathbf{1}} \mathbf{f}(\mathbf{t_i}) + \mathbf{f}(\mathbf{b})\right]$$
$$- \tfrac{\mathbf{b}-\mathbf{a}}{\mathbf{12}}\, \mathbf{h^2}\, \mathbf{f^{(2)}}(\xi), \quad \xi \in [\mathbf{a}, \mathbf{b}]$$

Preuve.

Pour $i = 1, 2, \cdots, N$

$$\int_{t_{i-1}}^{t_i} f(x)\, dx = \frac{h}{2} \left(f(t_{i-1}) + f(t_i)\right) - \frac{h^3}{12} f^{(2)}(\xi_i),$$
$$\xi_i \in [t_{i-1}, t_i]$$

On en déduit :

$$\int_a^b f(x)\, dx = \frac{h}{2} \left(f(a) + 2 \sum_{i=1}^{N-1} f(t_i) + f(b)\right)$$
$$- \tfrac{h^3}{12} \sum_{i=1}^{N} f^{(2)}(\xi_i)$$

On voit facilement que :

$$\min_{x \in [a,b]} f^{(2)}(x) \leq \frac{1}{N} \sum_{i=1}^{N} f^{(2)}(\xi_i) \leq \max_{x \in [a,b]} f^{(2)}(x)$$

Il existe donc $\xi \in [a, b]$ tel que :

$f^{(2)}(\xi) = \frac{1}{N} \sum_{i=1}^{N} f^{(2)}(\xi_i)$ en appliquant à la fonction continue $f^{(2)}$ le théorème des valeurs intermédiares.

Il vient alors : $\frac{h^3}{12} \sum_{i=1}^{N} f^{(2)}(\xi_i) = \dfrac{b-a}{12}\, h^2\, f^{(2)}(\xi)$

Remarque.

On appelle erreur la quantité : $-\frac{b-a}{12}\, h^2\, f^{(2)}(\xi)$

$\mid -\frac{b-a}{12}\, h^2\, f^{(2)}(\xi) \mid \leq K\, h^2$

avec $K = \frac{1}{12}\,(b-a) \max\limits_{x\in[a,b]} \mid f^{(2)}(x) \mid$

On dit que la méthode des Trapèzes est d'ordre 2.

– **Formule de Simpson composites :**

si $\mathbf{f} \in \mathcal{C}^{(4)}([\mathbf{a},\mathbf{b}])$ **avec** $\mathbf{h} = \frac{\mathbf{b-a}}{\mathbf{N}}$ **et** $\mathbf{M} = \frac{\mathbf{N}}{\mathbf{2}}$

$$\int_a^b f(x)\, dx = \frac{h}{3}\,[f(a) + 2\sum_{i=1}^{M-1} f(t_{2\,i}) + 4\sum_{i=1}^{M} f(t_{2\,i-1}) + f(b)] - \frac{b-a}{180}\, h^4\, f^{(4)}(\xi), \quad \xi \in [a,b]$$

Preuve.

Pour $M = \frac{N}{2}$ avec N pair, on fait la décomposition

$\displaystyle\int_a^b f(x)\, dx = \sum_{i=1}^{M} \int_{t_{2\,i-2}}^{t_{2\,i}} f(x)\, dx$ en M sous-intervalles de longueur $2\,h$.

On en déduit :

$$\int_a^b f(x)\, dx = \frac{h}{3}\sum_{i=1}^{M}(f(t_{2\,i-2}) + 4\ f(t_{2\,i-1}) + f(t_{2\,i})) - \frac{h^5}{90}\sum_{i=1}^{M} f^{(4)}(\xi_i)\ ,\ \xi_i \in [t_{2\,i-2}, t_{2\,i}]$$

$$= \frac{h}{3}\,[f(a) + 2\sum_{i=1}^{M-1} f(t_{2\,i}) + 4\sum_{i=1}^{M} f(t_{2\,i-1}) + f(b)] - \frac{h^5}{90}\sum_{i=1}^{M} f^{(4)}(\xi_i)$$

La continuité de $f^{(4)}$ entraîne l'existence de

$\xi \in [a,b]$ tel que

$$\frac{1}{M}\sum_{i=1}^{M} f^{(4)}(\xi_i) = f^{(4)}(\xi)$$

Le terme d'erreur s'écrit donc sous la forme :

$$-\frac{h^5}{90}\sum_{i=1}^{M} f^{(4)}(\xi_i) = \frac{M\,h^5}{90}\,f^{(4)}(\xi) = \frac{b-a}{180}\,h^4\,f^{(4)}(\xi)$$

Remarque.

$| -\frac{b-a}{180}\,h^4\,f^{(4)}(\xi)\,| \leq K\,h^4$

avec $K = \frac{1}{180}\,(b-a)\,\max_{x\in[a,b]}\,|\,f^{(4)}(x)\,|$

On dit que la méthode de Simpson est d'ordre 4.

Exemple :

$\int_0^4 e^x dx = e^4 - e^0 = 53.59815..$

Déterminer l'entier N pour être sûr d'obtenir une erreur de quadrature inférieure à 10^{-5}.

Trapèzes :

$|\,\frac{b-a}{12}\,h^2\,f^{(2)}(\xi)\,| = \frac{16}{3\,N^2}\;|\,e^{\xi}\,| \leq \frac{16}{3\,N^2}\,e^4 \leq 10^{-5}$

si $N \geq 5397$

Simpson :

$|\,\frac{b-a}{180}\,h^4\,f^{(4)}(\xi)\,| = \frac{256}{45\,N^4}\;|\,e^{\xi}\,| \leq \frac{256}{45\,N^4}\,e^4 \leq 10^{-5}$

si $N \geq 76$

En fait, pour obtenir une précision absolue de 10^{-5}, il faut prendre $N = 2674$ pour la méthode des Trapèzes et $N = 54$ pour la méthode de Simpson.

CHAPITRE 4

EQUATIONS NON-LINEAIRES

Notations

I : intervalle de $\mathbb{R}$.

$[a, b]$: intervalle fermé et borné de $\mathbb{R}$.

f : une fonction numérique réelle définie et continue sur I.

4.1 Séparation des racines

Problème : Déterminer les racines de l'équation $f(x) = 0$ sur I :

$$(\alpha \in I \ , \ f(\alpha) = 0)$$

Une méthode de résolution numérique comportera deux étapes

1. La séparation des racines ; cela consiste à déterminer un intervalle $[a, b] \subset I$ tel que $[a, b]$ contienne une racine α et une seule de l'équation $f(x) = 0$.
2. Le calcul successif des racines séparées.

La séparation des racines peut se faire essentiellement par :

1. Une technique de balayage de I :
   - On découpe I en n intervalles $[x_i, x_{i+1}]$
   - On calcule successivement $f(x_i)\, f(x_{i+1})$; si c'est strictement négatif, c'est qu'il existe un nombre impair des racines sur $]x_i, x_{i+1}[$
   - On recommence le découpage sur chaque intervalle où on a détecté au moins une racine jusqu'à "l'assurance" d'avoir isolé une racine.

2. ou graphiquement.

Exemple

$f(x) = e^x \, sin(x) - 1 \qquad I = [-\pi, \pi]$

L'intersection de $f_1(x) = sin(x)$ et $f_2(x) = e^{-x}$ permet de déceler deux racines α_1, α_2 avec
$\alpha_1 \in]0.5, 0.6[$ et $\alpha_2 \in]3, 3.1[$

4.2 Calcul d'une racine séparée

On se place dans la situation suivante :

Il existe un intervalle $[a, b] \subset I$ tel que α soit la seule racine de $f(x) = 0$ sur $]a, b[$ et $f(a)\,f(b) < 0$.

4.2.1 Méthode de Dichotomie (Algorithme)

Entrée : ε (la précision désirée).

N_0 (le nombre maximum d'itérations)

Sortie : valeur approchée de α ou un message d'échec.

Poser $n = 1$

Tant que $n \leq N_0$ et $\frac{b-a}{2} > \varepsilon$ **faire**

  Poser $p = \frac{a+b}{2}$

  Imprimer (n, a, b, p)

  Poser $n = n + 1$

  Si $f(a)\,f(p) > 0$ **Alors**

    Poser $a = p$

  Sinon

    Poser $b = p$

  Fsi

Ftque

Si $n > N_0$ **Alors**

Imprimer (la méthode a échouée après N_0 itérations)

Sinon

Fsi

Remarque

Dans un algorithme on impose toujours un nombre maximal d'itérations afin d'éviter les boucles sans fin causées soit par une convergence trop lente soit par une erreur sur les données initiales.

A chaque itération, l'algorithme construit un nouvel intervalle $[a_n, b_n]$, autour de la racine cherchée α, qui est de longueur égale à la moitié de la longueur de l'intervalle précédent, c'est-à-dire :

$$b_n - a_n = \frac{b_{n-1} - a_{n-1}}{2}$$

et donc on a :

$$b_n - a_n = \frac{b-a}{2^{n-1}}$$

Par conséquent, nous pouvons calculer le nombre maximal d'itérations pour obtenir $\frac{b_n - a_n}{2} \leq \varepsilon$

On a : $\frac{b-a}{2^n} \leq \varepsilon$, d'où l'on tire :

$$n \geq \frac{Log(b-a) - Log(\varepsilon)}{Log(2)}$$

Exemple

La fonction $f(x) = x^3 + 4\,x^2 - 10$ a au moins une racine dans l'intervalle [1,2] puisque $f(1) = -5$ et $f(2) = 14$.

La valeur de α à 9 chiffres significatifs est 1.36523001.

Si on voulait obtenir ce résultat il faudrait faire 29 itérations.

Il est interéssant de remarquer que l'approximation de α obtenue pour $n = 9$ est meilleure que pour $n = 12$.

n	p_n	$\mid p_n - p_{29} \mid$
9	1.36523438	0.00000437
12	1.36499023	0.00023978
29	1.36523001	-

Les avantages de cet algorithme sont la convergence assurée et une marge d'erreur sûre. Par contre il est relativement lent et il n'est pas toujours facile de localiser un intervalle sur lequel on a un changement de signe. Cet algorithme est souvent utilisé pour déterminer une approximation initiale à utiliser avec un algorithme plus rapide.

4.2.2 Méthode de Newton-Raphson

Soit $\overline{x} \in [a, b]$ une approximation de la racine α. On suppose donc que $\mid \overline{x} - \alpha \mid$ est petit et aussi que $f'(\overline{x}) \neq 0$.

Si $f \in \mathcal{C}^{(2)}([a, b])$, le développement de Taylor-Lagrange de f à l'ordre 2 au point $\overline{x}$ s'écrit :

$$f(x) = f(\overline{x}) + (x - \overline{x})\, f'(\overline{x}) + \frac{(x-\overline{x})^2}{2!}\, f"(\xi_x)$$

où ξ_x est un point entre x et $\overline{x}$.

En particulier, pour $x = \alpha$, on obtient :

$$f(\alpha) = 0 = f(\overline{x}) + (\alpha - \overline{x})\, f'(\overline{x}) + \frac{(\alpha-\overline{x})^2}{2!}\, f"(\xi_\alpha)$$

En négligeant l'infiniment petit $(\alpha - \overline{x})^2$ d'ordre 2, on peut s'attendre à ce que :

$$x^* = \overline{x} - \frac{f(\overline{x})}{f'(\overline{x})}$$

soit une meilleure approximation de α que $\overline{x}$.

Théorème

On suppose que $f \in \mathcal{C}^{(2)}([a, b])$ avec $\alpha \in]a, b[$.

Si α est une racine simple de l'équation $f(x) = 0$ alors il existe un réel $\theta > 0$ tel que pour tout $x_0 \in [\alpha - \theta, \alpha + \theta]$, l'itération de Newton :

$$x_{k+1} = x_k - \frac{f(x_k)}{f'(x_k)} \qquad k \geq 0$$

commençant en x_0 converge vers α.

Preuve.

$f'(\alpha) \neq 0$ puisque α est une racine simple.

Comme f' est continue sur [a,b], il existe alors $\theta_1 > 0$ tel que :

$\forall x \in [\alpha - \theta_1, \alpha + \theta_1] \subset [a, b]$, $f'(x) \neq 0$

Il suffit de montrer que la fonction g définie par

$g(x) = x - \frac{f(x)}{f'(x)}$

définie et continue sur $[\alpha - \theta_1, \alpha + \theta_1]$ vérifie les conditions suffisantes pour être une contraction.

Il est facile de vérifier que g est dérivable sur $[\alpha - \theta_1, \alpha + \theta_1]$, avec g' continue et définie par :

$$g'(x) = \frac{f(x)\, f"(x)}{[f'(x)]^2}$$

Puisque $g'(\alpha) = 0$, la continuité de g' assure l'existence d'un $\theta > 0,\ \theta < \theta_1$ tel que :

$\forall x \in [\alpha - \theta, \alpha + \theta]$, $\mid g'(x) \mid \leq K < 1$

Il reste à vérifier que g, définie sur $[\alpha - \theta, \alpha + \theta]$, est aussi à valeurs dans ce même intervalle

Or $\forall x \in [\alpha - \theta, \alpha + \theta]$ on a :

$\mid g(x) - \alpha \mid = \mid g(x) - g(\alpha) \mid = \mid g'(\xi) \mid \mid x - \alpha \mid$

$\leq K \ \mid x - \alpha \mid \leq \mid x - \alpha \mid \leq \theta$, ξ *est entre* x *et* α.

4.2.3 Algorithme de Newton-Raphson

Entrée : Une approximation initiale p

ε (la précision désirée).

N_0 (le nombre maximum d'itérations)

Sortie : valeur approchée de α ou un message d'échec.

Poser $n = 1$

Tant que $n \leq N_0$ et $| \frac{f(p)}{f'(p)} | > \varepsilon$ **faire**

Poser $p = p - \frac{f(p)}{f'(p)}$

Imprimer $(n, p, f(p))$

Poser $n = n + 1$

Ftque

Si $n > N_0$ **Alors**

Imprimer (la méthode a échouée après N_0

itérations)

Sinon

Fsi

Exemple

Considérons le même problème qu'en 2.1

c'est-à-dire : $f(x) = x^3 + 4\,x^2 - 10$

On a : $f'(x) = 3\,x^2 + 8\,x$

d'où : $p_{n+1} = p_n - \frac{p_n^3 + 4\,p_n^2 - 10}{3\,p_n^2 + 8\,p_n}$

pour $n = 5$ le résultat est précis à 9 chiffres significatifs

n	p_n	$f(p_n)$
1	1.5000000000	2.3750000000
2	1.3733333333	0.1343454815
3	1.3652620149	0.0005284612
4	1.3652300139	0.0000000083
5	1.3652300134	0.0000000000

4.2.4 Méthode de la sécante

Il est parfois ennuyeux dans l'utilisation de la méthode de Newton-Raphson d'avoir à calculer $f'(p_n)$. L'algorithme suivant peut être considéré comme une approximation de la méthode de Newton-Raphson.

Au lieu d'utiliser la tangente au point p_n nous allons utiliser la sécante passant par les points d'abscisses p_n et p_{n-1} pour en déduire p_{n+1}.

L'équation de la sécante s'écrit :

$$s(x) = f(p_n) + \frac{f(p_n)-f(p_{n-1})}{p_n-p_{n-1}} (x - p_n)$$

Si $s(p_{n+1}) = 0$, on tire :

$$p_{n+1} = p_n - \frac{p_n-p_{n-1}}{f(p_n)-f(p_{n-1})} f(p_n)$$

4.2.5 Algorithme de la sécante

Entrée : Deux approximations initiales p_0 et p_1

ε (la précision désirée).

N_0 (le nombre maximum d'itérations)

Sortie : valeur approchée de α ou un message d'échec.

Poser $n = 2$

$q_0 = f(p_0)$

$q_1 = f(p_1)$

Tant que $n \leq N_0 + 1$ et $\mid \frac{p_1-p_0}{q_1-q_0} q_1 \mid > \varepsilon$ **faire**

Poser $p = p_1 - \frac{p_1-p_0}{q_1-q_0} q_1$

Imprimer $(n, p, f(p))$

Poser $n = n + 1$

$p_0 = p_1$

$p_1 = p$

$q_0 = q_1$

$q_1 = f(p_1)$

Ftque

Si $n > N_0 + 1$ **Alors**

Imprimer (la méthode a échouée après N_0 itérations)

Sinon

Fsi

Remarque

Si p_n et p_{n-1} sont proches (on peut espérer que ce sera le cas si la méthode converge), on peut considérer que

$\frac{f(p_n - f(p_{n-1})}{p_n - p_{n-1}}$

est une approximation de $f'(p_n)$. Dans ce cas l'algorithme de la sćante est une approximation de l'algorithme de Newton.

Exemple

Considérons le même problème que précèdemment,

c'est-à-dire : $f(x) = x^3 + 4\,x^2 - 10$, avec comme point de départ $p_0 = 2$ et $p_1 = 1$

Nous constaterons que la convergence est moins bonne que celle obtenue par la méthode de Newton.

4.2.6 Méthode du point fixe

On remplace la résolution de l'équation

$$f(x) = 0 \quad (E_1)$$

par la résolution de l'équation

$$g(x) = x \quad (E_2)$$

Le choix de g doit être tel que :

1. la suite itérative définie par :

   $$x_0 \in [a, b]$$

   $$x_{n+1} = g(x_n) \quad n \geq 0$$

   soit convergente.

   Si g est continue la limite α de la suite $(x_n)_{n\geq 0}$ est alors solution de (E_2).

2. la solution de (E_2) soit aussi solution de (E_1).

Nous pouvons observer que la méthode de Newton s'écrit toujours :

$$p_{n+1} = g(p_n) \text{ où } g(x) = x - \frac{f(x)}{f'(x)}$$

Si la fonction g est continue et si l'algorithme converge, c'est-à-dire si p_n tend vers α, alors α vérifie : $\alpha = g(\alpha)$.

On dit que α est un **point fixe**.

Théorème

On considère une fonction g définie et continue sur [a,b], à valeurs dans [a,b]. On suppose que g est une contraction sur [a,b] : c'est-à-dire qu'il existe $K \in [0,1[$ tel que

$\forall (x, x') \in [a,b] \times [a,b]$:

$$\mid g(x) - g(x') \mid \leq K \mid x - x' \mid$$

Alors l'équation (E_2) admet une racine unique $\alpha \in [a,b]$ et on a

$\forall x_0 \in [a,b]$ la suite $(x_n)_{n\geq 0}$ définie par

$$x_{n+1} = g(x_n)$$

converge vers α et

$$\mid \alpha - x_n \mid \leq \frac{K^n}{1-K} \mid x_1 - x_0 \mid$$

Remarque

Ce théorème est un résultat de **convergence globale** car la convergence de la suite $(x_n)_{n\geq 0}$ vers α est assurée quel que soit l'initialisation x_0.

Preuve.

Montrons que la suite définie dans $i)$ converge vers une racine de (E_2).

$$\forall j \geq 0 \quad \mid x_{j+1} - x_j \mid = \mid g(x_j) - g(x_{j-1}) \mid$$

$$\leq K \mid x_j - x_{j-1} \mid \leq K^j \mid x_1 - x_0 \mid$$

pour $p \geq 1$ fixé, il vient :

$$\mid x_{n+p} - x_n \mid \leq \sum_{j=n}^{n+p-1} \mid x_{j+1} - x_j \mid$$

$$\leq \mid x_1 - x_0 \mid K^n \sum_{j=0}^{p-1} K^j \leq \mid x_1 - x_0 \mid K^n \sum_{j=0}^{\infty} K^j$$

soit $\mid x_{n+p} - x_n \mid \leq \frac{K^n}{1-K} \mid x_1 - x_0 \mid \qquad (\star)$

On en déduit que : $\lim\limits_{n \to +\infty} \mid x_{n+p} - x_n \mid = 0$, c'est-à-dire que $(x_n)_{n \geq 0}$ est une suite de cauchy dans $\mathbb{R}$ et par suite convergente vers $\alpha \in [a, b]$.

De la continuité de g, il découle que :

$$\begin{aligned} \alpha &= \lim_{n \to +\infty} x_{n+1} = \lim_{n \to +\infty} g(x_n) \\ &= g(\lim_{n \to +\infty} x_n) = g(\alpha) \end{aligned}$$

Montrons maintenant que (E_2) possède une racine sur [a,b], suposons l'existence de deux racines distinctes α et α', alors

$\mid \alpha - \alpha' \mid = \mid g(\alpha) - g(\alpha') \mid \leq K \mid \alpha - \alpha' \mid$

ce qui aboutit à la contradiction

$$(1 - K) \mid \alpha - \alpha' \mid \leq 0$$

En faisant tendre p vers ∞ dans l'inégalité $(\star)$ on trouve

$$\mid \alpha - x_n \mid \leq \frac{K^n}{1-K} \mid x_1 - x_0 \mid$$

Corollaire

On considère une fonction g définie et continue sur [a,b], à valeurs dans [a,b].

g est une contraction si les conditions suivantes sont vérifiées :

1. g est dérivable sur [a,b]
2. $\forall x \in [a, b]$, $\mid g'(x) \mid \leq K < 1$

Preuve

De la formule des accroissements finis, il vient que

$\forall (x, x') \in [a, b] \times [a, b]$, il existe $\xi \in]x, x'[$ tel que :

$$g(x) - g(x') = g'(\xi)\,(x - x')$$

d'où : $\forall (x, x') \in [a, b] \times [a, b]$,

$\mid g(x) - g(x') \mid = \mid g'(\xi) \mid \mid x - x' \mid \leq K \mid x - x' \mid$.

4.2.7 Algorithme du point fixe

But : Trouver une solution de $x = g(x)$

Entrée : Une approximation initiale p

ε (la précision désirée).

N_0 (le nombre maximum d'itérations)

Sortie : valeur approchée de α ou un message d'échec.

Poser $n = 1$

Tant que $n \leq N_0$ et $\mid g(p) - p \mid > \varepsilon$ **faire**

Poser $p = g(p)$

Imprimer $(n, p, g(p))$

Poser $n = n + 1$

Ftque

Si $n > N_0$ **Alors**

Imprimer (la méthode a échouée après N_0 itérations)

Sinon

Fsi

4.2.8 Exemple 1

$f(x) = x^3 + 4\,x^2 - 10\ ,\ [a, b] = [1, 2]$

On considère :

$$g_1(x) = 10 + x - 4\,x^2 - x^3$$
$$g_2(x) = \sqrt{\frac{10}{x} - 4\,x}$$
$$g_3(x) = \frac{1}{2}\sqrt{10 - x^3}$$
$$g_4(x) = \sqrt{\frac{10}{4+x}}$$

On pourra vérifier que g_3 et g_4 sont des bons choix pour résoudre l'équation $f(x) = 0$ ce qui n'est pas le cas de g_1 et g_2.

Exemple2

$f(x) = x^2 - 2\,x - 3$

L'équation $f(x) = 0$ admet deux racines qui sont -1 et 3.

On considère :

$$g(x) = \sqrt{2\,x + 3}$$

On utilise un estimé $x_0 = 4$, on obtient :

$$x_1 = \sqrt{11} \simeq 3.316$$

$$x_2 = \sqrt{9.632} \simeq 3.104$$

$$x_3 = \sqrt{9.208} \simeq 3.034$$

$$x_4 = \sqrt{9.068} \simeq 3.011$$

$$x_5 = \sqrt{9.022} \simeq 3.004$$

$(x_n)_{n\geq 0}$ apparaît comme une suite convergeant vers 3.

On considère :

$$g(x) = \tfrac{3}{x-2}$$

Si $x_0 = 4$, alors :

$$x_1 = 1.5$$

$$x_2 = -6$$

$$x_3 = -0.375$$

$$x_4 = -1.263$$

$$x_5 = -0.919$$

$$x_6 = -1.028$$

$$x_7 = -0.991$$

$$x_5 = -1.003$$

On voit que la suite $(x_n)_{n\geq 0}$ converge vers -1.

On considère :

$$g(x) = \frac{x^2-3}{2}$$

Pour $x_0 = 4$, on obtient :

$$x_1 = 6.5$$

$$x_2 = 19.635$$

$$x_3 = 191.0$$

Cette suite diverge.

2.2 Remarque

Il est parfois difficile de transformer $f(x) = 0$ en $x = g(x)$ par des transformations algèbriques simples.

Dans ces cas on pourra réecrire $f(x) = 0$ comme

$$x = x + k\,f(x) = g(x)$$

où k est une constante non nulle que l'on choisit

On choisit k afin que la dérivée de $g(x)$ soit telle que :

$$\mid g'(x) \mid = \mid 1 + k\,f'(x) \mid < 1$$

afin d'assurer la convergence.

(Prendre $k = \frac{1}{4}$ pour l'exemple2)

CHAPITRE 5

RESOLUTION NUMERIQUE DE PROBLEMES DIFFERENTIELS

5.1 Problème de Cauchy

Le but de ce chapitre est d'étudier des méthodes pour approximer les solutions $y(x)$ d'un problème de type

$$\frac{\mathbf{dy}}{\mathbf{dx}} = \mathbf{f}(\mathbf{x}, \mathbf{y}) \quad (1)$$

$$\mathbf{y}(\mathbf{x_0}) = \mathbf{y_0} \quad (2)$$

où la fonction $f(x, y)$ est une fonction connue de deux variables.

(1) détermine une famille de solutions $y = \Phi(x, c)$ (appelées courbes intégrales) dépendant d'un paramètre.

(2) permet de choisir le membre de cette famille de courbe passant par (x_0, y_0) (ceci, en déterminant une valeur particulière de c).

Du point de vue géométrique, l'équation (1) signifie que la pente de la courbe intégrale $y = \Phi(x, c)$ passant par un point (x, y) quelconque est donnée par $f(x, y)$.

Définition

On dit que la fonction $f : [a, b] \times \mathbb{R} \longrightarrow \mathbb{R}$ est lipschitzienne en y dans $[a, b] \times \mathbb{R}$ s'il existe $A > 0$ tel que :

$$\mid \mathbf{f}(\mathbf{x}, \mathbf{y}) - \mathbf{f}(\mathbf{x}, \mathbf{z}) \mid \leq \mathbf{A} \mid \mathbf{y} - \mathbf{z} \mid$$

$$\forall \mathbf{x} \in [\mathbf{a}, \mathbf{b}] \quad \forall \mathbf{y}, \mathbf{z} \in \mathbb{R}$$

A est appelée **constante de Lipschitz.**

Théorème (admis)

Soit le problème de Cauchy ((1),(2))

Si $f : [a, b] \times \mathbb{R} \longrightarrow \mathbb{R}$ vérifie les hypothèses :

1. f continue
2. f lipschitzienne en y

alors le problème de Cauchy ((1),(2)) admet une solution unique.

5.2 Approche numérique

On prend : $x_i = a + i\,h$, $i = 0, \cdots, N$ avec $h = \frac{b-a}{N}$ une subdivision de l'intervalle [a,b] en N intervalles égaux.

On cherche N nombres $y_1, y_2, \cdots, y_N$ où y_i est une valeur approchée de $y(x_i)$.

Puis on reliera ces points par interpolation pour définir une fonction y_h sur [a,b].

On va voir dans ce qui suit trois méthodes numériques permettant de calculer la solution approchée y_{i+1} au point x_{i+1} en utilisant celle au point x_i.

5.2.1 Méthode d'Euler

On considère que sur le petit intervalle
$[x_0, x_0 + h]$ la courbe n'est pas très éloignée de sa tangente en x_0

$$z(x) = y(x_0) + y'(x_0)\,(x - x_0)$$

Une bonne approximation de $y(x_1)$ est donnée par

$$y_1 = y_0 + h\,f(x_0, y_0)$$

On considère ensuite que $f(x_1, y_1)$ est une approximation de $y'(x_1)$ et sur l'intervalle $[x_1, x_2]$, on remplace la courbe par sa tangente approchée en x_1

$$\begin{aligned} z(x) &= y(x_1) + y'(x_1)\,(x - x_1) \\ &\simeq y_1 + f(x_1, y_1)\,(x - x_1) \end{aligned}$$

Une bonne approximation de $y(x_2)$ est donnée par

$$y_2 = y_1 + h\, f(x_1, y_1)$$

A la ième étape en approximant $y(x_i) \simeq y_i$, on obtient :

$$y(x_i + h) \simeq y_i + h\, f(x_i, y_i)$$

ce qui suggère de nouveau l'équation :

$$y_{i+1} = y_i + h\, f(x_i, y_i) \qquad (3)$$

appelée **équation aux différences pour la méthode d'Euler.**

Exemple

$$\mathbf{y' = -y + x + 1}$$

$$\mathbf{y(x_0) = 1}$$

La solution exacte est donnée par :

$$y(x) = x + e^{-x}$$

Prenons $n = 10$, $h = 0.1$, (3) devient :

$$y_{i+1} = 0.9\, y_i + 0.1\, x_i + 0.1 \qquad i = 0, 1, \cdots, 9$$

On obtient alors le tableau :

i	x_i	$y(x_i)$	y_i	$\mid y(x_i) - y_i \mid$
0	0	1.004837	1.000000	0.004837
1	0.1	1.018731	1.010000	0.008731
2	0.2	1.040818	1.029000	0.011818
3	0.3	1.070302	1.056100	0.014220
4	0.4	1.106531	1.090490	0.016041
5	0.5	1.148812	1.131441	0.017371
6	0.6	1.196585	1.178297	0.018288
7	0.7	1.249329	1.230467	0.018862
8	0.8	1.306570	1.287420	0.019149
9	0.9	1.367879	1.348678	0.019201

On peut remarquer que l'erreur croît légèrement lorsque x_i croît.

Convergence

Théorème

Si f vérifie les hypothèses :

1. $f \in \mathcal{C}^1([a,b] \times \mathbb{R})$

2. f est lipschitzienne en y :

$$| \mathbf{f}(\mathbf{x}, \mathbf{y}) - \mathbf{f}(\mathbf{x}, \mathbf{z}) | \leq \mathbf{K} \ | \mathbf{y} - \mathbf{z} |$$

$$\forall \mathbf{x} \in [\mathbf{a}, \mathbf{b}] \quad \forall \mathbf{y}, \mathbf{z} \in \mathbb{R} \ , \ K > 0$$

alors la méthode d'Euler converge.

Plus précisement, si on pose :

$$M = Max\{| \ y'(t) \ |, t \in [a,b]\},$$

on a la majoration :

$$| \ e_n \ | = | \ y_n - y(x_n) \ | \leq \frac{e^{K\,(b-a)} - 1}{K} \frac{M}{2} h$$

$$\text{et} \lim_{N \to +\infty} Max\{| \ e_n \ |, n = 1, \cdots, N\} = 0$$

Remarques

1. Le résultat du théorème précédent s'exprime sous la forme :

$$\mid e_n \mid \leq A\, h \ , \ A > 0 \text{ une constante}$$

   c'est-à-dire que la méthode d'Euler est du premier ordre.

2. Une méthode d'orde 1 ne converge pas assez vite pour donner des résultats pratiques intéressants.

Exemple

$y' = \frac{2}{x}\, y + x^2\, e^x \ , \ y(1) = 0 \ , \ x \in [1, 2]$

On veut déterminer le pas h pour que l'erreur soit inférieure à 0.1.

Puisque $\mid f(x, y) - f(x, z) \mid = \mid \frac{2}{x}\, (y - z) \mid$

$\forall x \in [1, 2], \ K = 2.$

D'autre part en résolvant l'équation on obtient :

$$y(x) = x^2\, e^x - e\, x^2,$$

d'où : $M = y'(2) = 4\, e\, (-1 + 2\, e)$

La borne devient donc :

$$\frac{e^2}{2}\, \frac{4\, e\, (-1+2\, e)}{2}\, h$$

Il suffira donc que $h \leq \frac{10^{-1}}{e^3\, (-1+2\, e)} \simeq 0.0011222$

5.2.2 Méthodes de Taylor

Une première façon d'améliorer la méthode d'Euler (au sens où l'erreur variera au moins comme h^2) consiste à utiliser un développement de Taylor jusqu'à l'ordre 2.

Si $y(x_i)$ est donné, on a :

$y(x_{i+1}) = y(x_i) + (x_{i+1} - x_i)\, y'(x_i) +$

$\frac{1}{2}\, (x_{i+1} - x_i)^2\, y^{(2)}(x_i) + \frac{1}{6}\, (x_{i+1} - x_i)^2\, y^{(3)}(\xi_i)$

où $\xi_i \in [x_i, x_{i+1}]$

Si $h = x_{i+1} - x_i$ est assez petit, on a donc l'égalité approchée

$y(x_{i+1}) \simeq y(x_i) + h\, y'(x_i) + \frac{h^2}{2}\, y^{(2)}(x_i)$

Puisque $y'(x) = f(x, y(x))$ on a :

$$y"(x) = \frac{\partial f}{\partial x}(x, y(x)) + \frac{\partial f}{\partial y}(x, y(x))\, y'(x)$$
$$= \frac{\partial f}{\partial x}(x, y(x)) + \frac{\partial f}{\partial y}(x, y(x))\, f(x, y(x))$$

Tout ceci suggère la méthode aux différences suivantes

$$\mathbf{y_{i+1} = y_i + h\, f(x_i, y_i) +}$$
$$\mathbf{\frac{h^2}{2}\, [\frac{\partial f}{\partial x}(x_i, y_i) + \frac{\partial f}{\partial y}(x_i, y_i)\, f(x_i, y_i)]} \quad (4)$$

appelée **méthode de Taylor d'ordre 2**

Remarques

1. On peut montrer que pour chaque x_i fixé, l'erreur varie comme h^2.
2. Cette méthode possède le désavantage qu'elle nécessite le calcul de $\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}$.
3. Cette méthode se généralise facilement, cependant les méthodes de Taylor nécessiteront le calcul de $\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}, \frac{\partial^2 f}{\partial x^2}, \frac{\partial^2 f}{\partial x \partial y}, \frac{\partial^2 f}{\partial y^2}., \cdots$, ce qui peut être très laborieux.

Exemple

$$\mathbf{y' = -y + x + 1}$$
$$\mathbf{y_0 = y(x_0) = 1}$$

Puisque $\frac{\partial f}{\partial x} = 1, \frac{\partial f}{\partial y} = -1$ l'équation aux différences s'écrira :

$$y_{i+1} = y_i + h\,(-y_i + x_i + 1) + \frac{h^2}{2}\,[1 - (-y_i + x_i + 1)]$$
$$y_{i+1} = y_i + h\,[(\frac{h}{2} - 1)\, y_i + (1 - \frac{h}{2})\, x_i + 1]$$

Prenons $h = 0.1$ on obteint alors le tableau suivant :

y_i	erreur
1.0190251	2.942×10^{-4}
1.07082	4.820×10^{-4}
1.149404	5.924×10^{-4}
1.2949976	6.470×10^{-4}

5.2.3 Méthode de Runge-Kutta

On voudrait conserver l'avantage des méthodes d'ordre supérieur mais corriger les inconvénients dûs au calcul des dérivées partielles de f.

Ces différentes méthodes sont basées sur la formule de Taylor à plusieurs variables :

$$f(x,y) = f(x_0,y_0) + [(x-x_0)\,\tfrac{\partial f}{\partial x} + (y-y_0)\,\tfrac{\partial f}{\partial y}] +$$
$$\tfrac{1}{2!}\,[(x-x_0)^2\,\tfrac{\partial^2 f}{\partial x^2} + 2\,(x-x_0)\,(y-y_0)\,\tfrac{\partial^2 f}{\partial x \partial y} +$$
$$(x-x_0)^2\,\tfrac{\partial^2 f}{\partial x^2}] + \cdots + \tfrac{1}{(n+1)!}\sum_{j=0}^{n+1} C_{n+1}^{j}\,[(x-x_0)^{n+1-j}$$
$$(y-y_0)^j\,\tfrac{\partial^{n+1} f}{\partial x^{n+1-j}\,\partial y^j}(\xi,\eta)] \qquad (5)$$

où $\xi \in (x, x_0)$, $\eta \in (y, y_0)$

Parmi toutes ces méthodes, la plus utilisée est celle d'ordre 4 mais les calculs menant à l'algorithme sont très laborieux.

Illustrons l'idée pour la méthode de Runge-Kutta d'ordre 2.

L'idée est d'essayer de remplacer le terme

$$f(x_i,y_i) + \tfrac{h}{2}\,[\tfrac{\partial f}{\partial x}(x_i,y_i) + \tfrac{\partial f}{\partial y}(x_i,y_i)\,f(x_i,y_i)]$$

dans la formule (4) par une expression de type

$$a\,f(x_i+\alpha, y_i+\beta)$$

en utilisant (5) on obtient :

$$a\,f(x_i+\alpha, y_i+\beta) = a\,f(x_i,y_i) + a\,\alpha\,\tfrac{\partial f}{\partial x}(x_i,y_i) +$$
$$a\,\beta\,\tfrac{\partial f}{\partial y}(x_i,y_i) + a\,R_2(\xi_i,\eta_i)$$

avec $\xi_i \in (x_i, x_i + \alpha)$, $\eta_i \in (y_i, y_i + \beta)$

En identifiant les coefficients de f , $\frac{\partial f}{\partial x}$, $\frac{\partial f}{\partial y}$ on voit que l'on doit choisir :

$$a = 1 \ \ , \ \ a\,\alpha = \tfrac{h}{2} \ \ , \ \ a\,\beta = \tfrac{h}{2}\, f(x_i, y_i)$$

c'est-à-dire :

$$a = 1 \ \ , \ \ \alpha = \tfrac{h}{2} \ \ , \ \ \beta = \tfrac{h}{2}\, f(x_i, y_i)$$

En substituant dans (4) on obtient donc la formule aux différences :

$$\mathbf{y_{i+1} = y_i + h\, f(x_i + \tfrac{h}{2}, y_i + \tfrac{h}{2}\, f(x_i, y_i))}$$

d'où l'algorithme suivant :

Méthode de Runge-Kutta d'ordre 2

Soit $N = \frac{x_N - x_0}{h}$=nombre de pas nécessaire

Poser $i = 0$

Tant que $i < N$ **faire**

$k = h\,f(x_i, y_i)$

$y_{i+1} = y_i + h\,f(x_i + \frac{h}{2}, y_i + \frac{k}{2})$

$x_{i+1} = x_i + h$

$i = i + 1$

Ftque

Ceci nous amène à la méthode de Runge-Kutta d'ordre 4 pour laquelle nous donnons directement l'algorithme :

Méthode de Runge-Kutta d'ordre 4

Soit $N = \frac{x_N - x_0}{h}$=nombre de pas nécessaire

Poser $i = 0$

Tant que $i < N$ **faire**

$k_1 = h\,f(x_i, y_i)$

$k_2 = h\,f(x_i + \frac{h}{2}, y_i + \frac{1}{2}\,k_1)$

$k_3 = h\,f(x_i + \frac{h}{2}, y_i + \frac{1}{2}\,k_2)$

$k_4 = h\,f(x_i + h, y_i + k_3)$

$y_{i+1} = y_i + \frac{1}{6}\,(k_1 + 2\,k_2 + 2\,k_3 + k_4)$

$x_{i+1} = x_i + h$

$i = i + 1$

Ftque

Commentaire

Les méthodes de Runge-Kutta sont faciles à utiliser, cependant elles sont relativement coûteuses car elles nécessiteront plusieurs évaluations de $f(x, y)$ à chaque pas.

Exemple

1. $y' = -y + x^2 + 1 \ \ , \ \ 0 < x \leq 1 \ \ , \ \ y(0) = 1$

Solution exacte est : $y(x) = -2\,e^{-x} + x^2 - 2\,x + 3$.

Utiliser la méthode de Runge-Kutta d'ordre 2 avec $h = 0.1$, $N = \frac{1-0}{0.1} = 10$

x_i	$y(x_i)$	y_i	erreur : $\mid y_i - y_i \mid$
0	1.00000	1.00000	0
0.1	1.00033	1.00025	0.0000751639
0.2	1.00254	1.00243	0.0001122438
0.3	1.00836	1.00825	0.0001178024
0.4	1.01936	1.01926	0.0000974985
0.5	1.03694	1.03688	0.0000562001
0.6	1.06238	1.06238	0.0000019171
0.7	1.09683	1.09690	0.0000732812
0.8	1.14134	1.14150	0.0001548478
0.9	1.19686	1.19710	0.0002440317
1.0	1.26424	1.26458	0.0003386469

2. $y' = -y + x + 1 \ \ , \ \ 0 \leq x \leq 1 \ \ , \ \ y(0) = 1$

Solution exacte est : $y(x) = -2\,e^{-x} + x^2 - 2\,x + 3$.

Utiliser la méthode de Runge-Kutta d'ordre 4 avec $h = 0.1$, $N = \frac{1-0}{0.1} = 10$

On peut comparer les résultats avec ceux obtenus précèdemment pour voir l'augmentation de la précision.

x_i	$y(x_i)$	y_i	erreur : $\mid y_i - y_i \mid$
0	1.0000000000	1.0000000000	0.0000000000
0.1	1.0048374180	1.0048375000	0.0000000820
0.2	1.0187307531	1.0187309014	0.0000001483
0.3	1.0408182207	1.0408184220	0.0000002013
0.4	1.0703200460	1.0703202889	0.0000002429
0.5	1.1065306597	1.1065309344	0.0000002747
0.6	1.1488116361	1.1488119344	0.0000002983
0.7	1.1965853038	1.1965856187	0.0000003149
0.8	1.2493289641	1.2493292897	0.0000003256
0.9	1.3065696597	1.3065699912	0.0000003315
1.0	1.3678794412	1.3678797744	0.0000003332

RESOLUTION DES SYSTEMES LINEAIRES

Introduction

La résolution de grands systèmes (linéaires ou non-linéaires) est pratique courante de nos jours, spécialement en génie mécanique, en génie électrique, et de façon générale, dans tous les domaines où l'on s'intéresse à la résolution numérique d'équations aux dérivées partielles.

Rappel sur les systèmes linéaires

Un système de n équations à n inconnues peut toujours s'écrire sous la forme :

$$A\,x = b$$

où $A = (a_{i,j})_{1 \leq i,j \leq n}$ est une matrice $n \times n$
et x et b sont des vecteurs colonnes de dimension n.

Méthode de résolution

La méthode de résolution la plus étudiée (et une des plus utilisées) s'appelle méthode d'élimination de Gauss. Elle a pour but de remplacer le système $A\,x = b$ par un système triangulaire supérieur avec des 1 dans la diagonale. C'est-à-dire de la forme :

$$\begin{pmatrix} 1 & \tilde{a}_{1,2} & . & . & . & \tilde{a}_{1,n} \\ & 1 & \tilde{a}_{2,3} & . & . & \tilde{a}_{2,n} \\ & & . & . & . & . \\ & & & . & . & . \\ & & & & 1 & \tilde{a}_{n-1,n} \\ & & & & & 1 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ . \\ . \\ x_{n-1} \\ x_n \end{pmatrix} = \begin{pmatrix} \tilde{b}_1 \\ \tilde{b}_2 \\ . \\ . \\ \tilde{b}_{n-1} \\ \tilde{b}_n \end{pmatrix}$$

On en déduit alors facilement les composantes du vecteur solution x :

$x_n = \tilde{b}_n$
$x_i = \tilde{b}_i - \sum_{j=i+1}^{n} \tilde{a}_{i,j}\, x_j \quad pour \;\; i = n-1, \cdots, 1$

Elimination de Gauss sur un exemple

$$\begin{cases} x_1 + x_2 = 3 \\ 2\,x_1 + x_2 - x_3 = 1 \\ 3\,x_1 - x_2 - x_3 = -2 \end{cases}$$

qui s'écrit :

$$\begin{pmatrix} 1 & 1 & 0 \\ 2 & 1 & -1 \\ 3 & -1 & -1 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} 3 \\ 1 \\ -2 \end{pmatrix}$$

- **Elimination de la première inconnue**

Eliminons x_1 dans les deux équations 2 et 3. Pour ce faire on multiplie la première équation par $a_{i,1}$ et on soustrait à ces deux équations

$$\begin{pmatrix} 1 & 1 & 0 \\ 0 & -1 & -1 \\ 0 & -4 & -1 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} 3 \\ -5 \\ -11 \end{pmatrix}$$

- **Elimination de la deuxième inconnue**

On divise la deuxième équation par le 2ème pivot

$$\begin{pmatrix} 1 & 1 & 0 \\ 0 & 1 & 1 \\ 0 & -4 & -1 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} 3 \\ 5 \\ -11 \end{pmatrix}$$

Eliminons x_2 dans la dernière équation. Puis divisons l'équation ainsi obtenue par le 3ème pivot

$$\begin{pmatrix} 1 & 1 & 0 \\ 0 & 1 & 1 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} 3 \\ 5 \\ 3 \end{pmatrix}$$

Finalement on trouve :

$$x_3 = 3 \qquad x_2 = 2 \qquad x_1 = 1$$

Algorithme : Elimination de Gauss

Pour $i = 1$ jusqu'à $n - 1$ faire

Pour $j = i + 1$ jusqu'à n faire

$a_{i,j} \leftarrow \frac{a_{i,j}}{a_{i,i}}$

Fpour

$b_i \leftarrow \frac{b_i}{a_{i,i}}$

Pour $k = i + 1$ jusqu'à n faire

Pour $j = i + 1$ jusqu'à n faire

$a_{k,j} \leftarrow a_{k,j} - a_{k,i}\, a_{i,j}$

Fpour

$b_k \leftarrow b_k - a_{k,i}\, b_i$

Fpour

Fpour

$b_n \leftarrow \frac{b_n}{a_{n,n}}$

Cette première phase de la méthode de Gauss est la triangularisation. Il reste maintenant à résoudre le système triangulaire supérieur obtenu après la triangularisation.

$x_n \leftarrow b_n$

Pour $i = n - 1$ jusqu'à 1 pas -1 faire

$$x_i \leftarrow b_i - \sum_{j=i+1}^{n} a_{i,j}\, x_j$$

Fpour

Notons toutefois que notre algorithme ne peut être exécuté jusqu'à la fin que si les pivots successifs sont tous non nuls.

Nous pouvons nous demander s'il existe une relation entre la matrice de départ et les matrices triangulaires obtenues. Ce lien existe :

Si U et L désignent les matrices triangulaires obtenues après triangularisation :

$$L = \begin{pmatrix} l_{1,1} & & & & & \\ l_{2,1} & l_{2,2} & & & & \\ . & . & . & & & \\ . & . & . & . & & \\ l_{n-1,1} & l_{n-1,2} & . & . & l_{n-1,n-1} & \\ l_{n,1} & l_{n,2} & . & . & l_{n,n-1} & l_{n,n} \end{pmatrix}$$

$$U = \begin{pmatrix} 1 & u_{1,2} & . & . & . & u_{1,n} \\ & 1 & u_{2,3} & . & . & u_{2,n} \\ & & . & . & . & . \\ & & & . & . & . \\ & & & & 1 & u_{n-1,n} \\ & & & & & 1 \end{pmatrix}$$

(et si l'algorithme d'élimination n'exige pas d'échange de lignes), on a :

$$A = L\,U$$

On dit dans ce cas que l'ona décomposé (ou factorisé) A en un produit $L\,U$.

Nous ne démontrerons pas cette proposition. Nous nous contenterons de la vérifier sur notre exemple :

$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 2 & -1 & 0 & 0 \\ 3 & -4 & 3 & 0 \\ -1 & 3 & 0 & -13 \end{pmatrix} \begin{pmatrix} 1 & 1 & 0 & 3 \\ 0 & 1 & 1 & 5 \\ 0 & 0 & 1 & \frac{13}{3} \\ 0 & 0 & 0 & 1 \end{pmatrix} = \begin{pmatrix} 1 & 1 & 0 & 3 \\ 2 & 1 & -1 & 1 \\ 3 & -1 & -1 & 2 \\ -1 & 2 & 3 & -1 \end{pmatrix}$$

Il y a une classe importante de matrices pour lesquelles l'élimination peut toujours s'opérer sans échange de lignes (c'est-à-dire le pivot $a_{j,j}$ n'est jamais nul pendant l'algorithme d'élimination). Ce sont les matrices à diagonale strictement dominante.

Définition

Une matrice A est dite à diagonale strictement dominante si pour $i = 1, 2, \cdots, n$ l'inégalité stricte

$$|a_{i,i}| > \sum_{j=1_{j \neq i}}^{n} |a_{i,j}|$$

est vérifiée.

Nombre d'opération pour l'élimination de Gauss

Pour évaluer la rapidité d'un algorithme, il est important de connaître le nombre d'opération qu'il nécessite.
On montre que l'algorithme d'élimination de Gauss fait appel à :

- $\frac{n(n^2-1)}{3}$ additions et autant de multiplications
- $\frac{n(n+1)}{2}$ divisions

La résolution du système triangulaire supérieur, quant à elle, nécessite $\frac{n(n-1)}{2}$ additions et autant de multiplications.

Au total la résolution d'un système de n équations linéaires par la méthode de Gauss demande :

- $\frac{n(n-1)(2n+5)}{6}$ additions et autant de multiplications
- $\frac{n(n+1)}{2}$ divisions

Remarques

1. On voit que, pour n grand, le nombre d'additions et de multiplications est de l'ordre de $\frac{n^3}{3}$. Ainsi si l'on multiplie par 2 la dimension d'un système il faudra 8 fois plus de temps pour le résoudre.
2. Si l'élément diagonal $a_{i,i}$ est nul, on cherche, dans la même colonne, un élément d'indice plus grand non nul, puis on échange les lignes correspondantes. Si ceci est impossible, le système est singulier.
3. On est parfois amené, pour des raisons de stabilité numérique, à effectuer des échanges de lignes même si $a_{i,i}$ est non nul. Ceci conduit à des stratégies dites de pivots.

Elimination de Gauss avec changement de pivots

Exemple

On veut résoudre :

$$\begin{pmatrix} 0 & 1 & 3 \\ 5 & 2 & 3 \\ 6 & 8 & 1 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} 1 \\ 4 \\ 1 \end{pmatrix}$$

Echange de lignes

Dans ce cas on ne peut pas directement appliquer l'algorithme d'élimination de Gauss puisque le 1er pivot est nul.

Toutefois, on peut échanger deux lignes pour obtenir un premier pivot non nul.

On choisit comme premier pivot le plus grand coefficient de la 1ère colonne. Pour cela on échange la 1ère et la 3ème ligne.

$$\begin{pmatrix} 6 & 8 & 1 \\ 5 & 2 & 3 \\ 0 & 1 & 3 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} 1 \\ 4 \\ 1 \end{pmatrix}$$

Maintenant on peut appliquer l'algorithme d'élimination de Gauss

- **Elimination de la première inconnue**

  La première opération consiste à diviser cette équation par $a_{1,1} = 6$ encore appelé premier pivot

$$\begin{pmatrix} 1 & \frac{4}{3} & \frac{1}{6} \\ 5 & 2 & 3 \\ 0 & 1 & 3 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} \frac{1}{6} \\ 4 \\ 1 \end{pmatrix}$$

  Eliminons l'inconnue x_1 dans les deux équations 2 et 3. Pour ce faire on multiplie la première équation par $a_{i,1}$ et on soustrait à ces deux équations

$$\begin{pmatrix} 1 & \frac{4}{3} & \frac{1}{6} \\ 0 & -\frac{14}{3} & \frac{13}{6} \\ 0 & 1 & 3 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} \frac{1}{6} \\ \frac{19}{6} \\ 1 \end{pmatrix}$$

- **Elimination de la deuxième inconnue**

  On divise la deuxième équation par le 2ème pivot

$$\begin{pmatrix} 1 & \frac{4}{3} & \frac{1}{6} \\ 0 & 1 & -\frac{13}{28} \\ 0 & 1 & 3 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} \frac{1}{6} \\ -\frac{19}{28} \\ 1 \end{pmatrix}$$

  Eliminons x_2 dans la dernière équation. Puis divisons l'équation ainsi obtenue par le 3ème pivot

$$\begin{pmatrix} 1 & \frac{4}{3} & \frac{1}{6} \\ 0 & 1 & -\frac{13}{28} \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} \frac{1}{6} \\ -\frac{19}{28} \\ \frac{47}{97} \end{pmatrix}$$

  Finalement on trouve :

$$x_3 = \frac{47}{97} \qquad x_2 = -\frac{44}{97} \qquad x_1 = \frac{67}{97}$$

Algorithme : Elimination de Gauss avec choix du pivot

Pour $i = 1$ jusqu'à $n-1$ faire

  Pivotage partiel

  Pour $j = i+1$ jusqu'à n faire

    $a_{i,j} \leftarrow \frac{a_{i,j}}{a_{i,i}}$

  Fpour

  $b_i \leftarrow \frac{b_i}{a_{i,i}}$

  Pour $k = i+1$ jusqu'à n faire

    Pour $j = i+1$ jusqu'à n faire

      $a_{k,j} \leftarrow a_{k,j} - a_{k,i}\, a_{i,j}$

    Fpour

    $b_k \leftarrow b_k - a_{k,i}\, b_i$

Fpour

Fpour

$b_n \leftarrow \frac{b_n}{a_{n,n}}$

La procédure du pivotage partiel, introduite dans la 1ère ligne de l'algorithme précédent s'écrit :

$k \leftarrow i$

$m \leftarrow abs(a_{i,i})$

Pour $j = i + 1$ jusqu'à n faire

$s \leftarrow abs(a_{j,i})$

si $m < s$ alors

$k \leftarrow j$ et $m \leftarrow s$

sinon

Fsi

Fpour

Si $k \neq i$ alors

Pour $j = i$ jusqu'à n faire

$t \leftarrow a_{i,j}$

$a_{i,j} \leftarrow a_{k,j}$

$a_{k,j} \leftarrow t$

Fpour

$t \leftarrow b_i$

$b_i \leftarrow b_k$

$b_k \leftarrow t$

sinon

Fsi

Factorisation

Dans le cas sans pivotage, si l'on doit résoudre un système où le membre de droite change, il y a intérêt à effectuer la réduction à la forme triangulaire une fois pour toutes.

En effet, si $A = L\,U$ on peut résoudre : $A\,x = b$ en résolvant :

$$L\,z = b$$

$$U\,x = z$$

Les systèmes étant triangulaires, la résolution ne nécessite que l'exécution d'une remontée et d'une descente triangulaire.

Remarque

Lorsque la matrice A est symétrique définie positive il y a intérêt à utiliser la méthode de Cholesky qui est une variante de la méthode de Gauss.
La méthode de Cholesky consiste à effectuer
la décomposition $A = L\,L^t$, c'est-à-dire que
$U = L^t$. On effectue alors deux fois moins d'opérations arithmétiques qu'avec la méthode de Gauss.

Méthode de Crout

Nous allons montrer sur un cas particulier que, l'on peut parfois factoriser directement une matrice A, en tenant compte de certaines structures particulières.

Soit

$$A = \begin{pmatrix} a_{1,1} & a_{1,2} & & & & & \\ a_{2,1} & a_{2,2} & a_{2,3} & & & & \\ & a_{3,2} & a_{3,3} & a_{3,4} & & & \\ & & . & . & . & & \\ & & & & a_{n-1,n-2} & a_{n-1,n-1} & a_{n-1,n} \\ & & & & & a_{n,n-1} & a_{n,n} \end{pmatrix}$$

une matrice tridiagonale. Nous cherchons des facteurs L et U de la forme :

$$L = \begin{pmatrix} l_{1,1} & & & & & \\ l_{2,1} & l_{2,2} & & & & \\ & l_{3,2} & l_{3,3} & & & \\ & & \cdot & \cdot & & \\ & & & l_{n-1,n-2} & l_{n-1,n-1} & \\ & & & & l_{n,n-1} & l_{n,n} \end{pmatrix}$$

$$U = \begin{pmatrix} 1 & u_{1,2} & & & & \\ & 1 & u_{2,3} & & & \\ & & 1 & u_{3,4} & & \\ & & & \cdot & \cdot & \\ & & & & 1 & u_{n-1,n} \\ & & & & & 1 \end{pmatrix}$$

La multiplication matricielle conduit aux équations:

1. $l_{1,1} = a_{1,1}$ (ligne 1 par colonne 1)
2. $l_{j,j-1}\, u_{j-1,j} + l_{j,j} = a_{j,j}$ (ligne j par colonne j)
3. $l_{j+1,j} = a_{j+1,j}$ (ligne j+1 par colonne j)
4. $l_{j,j}\, u_{j,j+1} = a_{j,j+1}$ (ligne j par colonne j+1)

Ceci suggère l'approche suivante :

Pour $j = 1, 2, \cdots, n$

- Déterminer la colonne j de L à l'aide de (b) ((a) si j=1) puis de (c).
- Déterminer la rangée j de U à l'aide de (d).

Si on se réfère à (c) on voit que la sous-diagonale de L coincide avec celle de A. Il suffit donc de stocker la diagonale de L et la sur-diagonale de U, ce que l'on fait, bien sûr, dans deux vecteurs.